SYNCHRONIZATION OF PLANNING INFORMATION IN A HIGH AVAILABILITY PLANNING AND SCHEDULING ARCHITECTURE

CROSS-REFERENCE TO RELATED APPLICATION

This application is related to U.S. Application No. 09/____,____ entitled "High Availability Planning and Scheduling Architecture," which was filed on April 13, 2001 by Subhasis Ojha, et al., Attorney's Docket 020431.0864.

TECHNICAL FIELD OF THE INVENTION

This invention relates to the field of planning and scheduling, and more particularly to synchronization of planning information in a high availability planning and scheduling architecture.

BACKGROUND OF THE INVENTION

Since product manufacturing and distribution facilities have a finite production and inventory capacities, planning for and managing customer orders and other requests is a fundamental activity performed by most manufacturing and distribution organizations. To better meet customer demand, a manufacturer may typically manufacture products before receiving customer orders. This production is typically based on forecasts of future customer demand. The supply of a product that is produced based on a demand forecast may be referred to as "available-to-promise" (ATP) supply. ATP supply consists of quantities of one or more products each having an associated date on which the products are scheduled to be available for delivery to a customer.

Once future demand has been forecasted, a plan to fulfill this demand (for example, using ATP supply) can be formulated by an appropriate planning engine. When an actual customer request is received, one or more promises may be made to the customer by a demand fulfillment engine regarding fulfillment of the request. The demand fulfillment engine typically has access to ATP supply information and can promise ATP supply to the customers to meet their demand. If a manufacturer or distributor is to satisfy its customers, it is often important that this demand fulfillment function be provided to the customers on a continuous basis without any interruptions in service. If uninterrupted demand fulfillment is not provided, a product manufacturer or distributor risks losing its customers to a competitor.

SUMMARY OF THE INVENTION

According to the present invention, disadvantages and problems associated with previous planning and scheduling systems and methods have been substantially reduced or eliminated.

According to one embodiment of the present invention, a method for synchronizing planning information in a high availability planning and scheduling architecture includes processing requests from one or more external systems using an advanced planning and scheduling (APS) engine included in a first primary high availability (HA) system. The processing of requests includes modifying planning information stored in memory of the first primary HA system according to the requests. The method also includes storing change information reflecting the modifications to the planning information in a database and extracting the change information from the database at an extraction time. Furthermore, the method includes updating the planning information using the extracted change information and storing the updated planning information in memory of a second primary HA system. In addition, the method includes identifying requests that were processed by the first primary HA system after the extraction time and updating the planning information stored in memory of the second primary HA system to account for these requests. The method also includes replacing the first primary HA system with the second primary HA system.

Particular embodiments of the present invention may provide one or more important technical advantages. For example, embodiments of the present invention provide an architecture that provides fault tolerant, real-time communication between a number of APS engines and one or more external systems. Certain embodiments of the present invention provide multiple APS engines that are in communication with one another such that if one APS engine fails or otherwise becomes unavailable, another APS engine can seamlessly take over for the unavailable APS engine. Furthermore, the multiple APS engines allow for load balancing through assignment of different types of requests to different APS engines and/or allocation of multiple requests of a particular type between multiple APS engines.

Particular embodiments also support the transformation of requests from external systems as appropriate for a particular APS engine and/or the transformation

of responses from an APS engine as appropriate for a particular external system. Therefore, the manner in which the APS engines and the external systems output information may not need to be specially configured before these components are included in the architecture of the present invention. Furthermore, particular embodiments of the present invention provide a mechanism by which the planning information stored in the APS engines (which is used to respond to requests from external system) may be updated while still providing uninterrupted service to the external systems.

Other important technical advantages are readily apparent to those skilled in the art from the following figures, description and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

To provide a more complete understanding of the present invention and the features and advantages thereof, reference is made to the following description taken in conjunction with the accompanying drawings, in which:

FIGURE 1 illustrates an exemplary high availability planning and scheduling architecture;

FIGURES 2A and 2B illustrates an exemplary method of processing product orders;

FIGURE 3 illustrates an exemplary method of processing product inquiries;

FIGURE 4 illustrates an exemplary operating and replacement HA system groups within an exemplary high availability planning and scheduling architecture; and

FIGURE 5 illustrates an exemplary method of updating and synchronizing planning information between an operating HA system and a replacement HA system.

DETAILED DESCRIPTION OF THE INVENTION

FIGURE 1 illustrates an exemplary high availability (HA) planning and scheduling architecture 10. Architecture 10 includes one or more HA systems 20 that communicate with one or more external systems 40 to provide advanced planning and scheduling (APS) services to users of external systems 40. HA systems 20 are coupled and share information in a manner that allows them to collectively provide fault-tolerant, real-time APS services to users of external system 40. In a particular embodiment, a primary HA system 20a provides primary services to external systems 40, and one or more redundant secondary HA systems 20b-20n provide secondary services to external systems 40 (for example, to alleviate the load on primary HA system 20a). One or more secondary HA systems 20b-20n may provide the primary services to external systems 40 (and thus assume the role of the primary HA system 20a) if the operation of primary HA system 20a is interrupted. As described below, primary HA system 20a communicates appropriate information to secondary HA systems 20b-20n as system 20a is providing the primary services to external system 40 such that secondary HA systems 20a-20n may seamlessly replace primary HA system 20a in the event of its failure or other unavailability.

In addition to providing redundant HA systems 20 for fault protection, the architecture 10 also provides for a controlled changeover between different groups of HA systems 20. Each HA system group includes a primary HA system 20a and one or more secondary HA systems 20b-20n. As described below, an operating HA system group may be replaced by a replacement HA system group so that ATP supply, customer order, or other information stored by the operating HA system group may be updated (or for any other appropriate reason). To accomplish such a changeover, architecture 10 includes a database system 60 that stores information regarding the state of the operating HA systems 20 and services that have been and are being performed by the operating HA systems 20. Therefore, one or more replacement HA systems 20 in the replacement HA system group may access database system 60 to obtain information necessary to perform the changeover. The operating and replacement HA systems 20 may then communicate to effect the changeover.

Although "high availability" is discussed in describing architecture 10 and systems 20, architecture 10 and systems 20 need not necessarily provide a particular level of availability (e.g., "five nines" or 99.999 percent availability). The present invention contemplates any suitable level of substantially continuous availability being provided. Moreover, the present invention is intended to encompass architectures and systems that seek to achieve such substantially continuous availability, whether or not it is actually achieved in a particular implementation.

HA systems 20 (both primary and secondary) each include APS engine 22 that receives requests from one or more external systems 40 and responds to the requests in an appropriate manner. APS engine 22 may provide any appropriate planning, scheduling, logistics, forecasting, fulfillment, management, and/or other related services to external systems 40. In the exemplary embodiments described below, HA systems 20 each include a demand fulfillment (DF) engine 22 that receives orders for products or other items from external ordering systems 40, evaluates the available supply of the items that may be used to fulfill the orders (the amounts of the items that are "available to promise" or ATP), and returns promises to external ordering systems 40 regarding the fulfillment of the orders. However, it will be understood from the following description that architecture 10 of the present invention may be used with any other appropriate type of APS engine 22 for which continuous or at least substantially continuous availability is desired. In such embodiments, DF engine 22 may be replaced in HA systems 20 by another suitable APS engine 22.

Each HA system 20 includes components in addition to APS engine 22 that are used to provide services to external systems 40. A message bus 50 enables the communication of messages between external ordering systems 40, HA systems 20, and database system 60. Message bus 50 may be implemented using any appropriate wireline or wireless communication technique. For example, in a particular embodiment, message bus 50 may include a portion of the Internet. Each external system 40 is coupled to one or more gateways (or HA clients) 52 that serve as an interface between external systems 40 and message bus 50. In a similar manner, HA servers 24 serve as an interface between the other components of HA systems 20 and message bus 50. A similar HA server 62 serves as an interface between database 64 of database system 60 and message bus 50. A message daemon or other appropriate

messaging manager 54 directs messages communicated over message bus 50 to one or more appropriate HA servers 24 and notifies HA servers 24 of the existence of messages. For purposes of simplicity, much of the description below assumes that HA servers 24 receive messages directly from and communicate messages directly to gateways 52 using message bus 50. However, it should be understood that this communication may be enabled using message manager 54 (and possibly an associated daemon or other manager at each HA system 20) to appropriately direct messages from external systems 40 to HA systems 20 and vice versa.

HA servers 24 receive and queue incoming messages from message bus 50 and perform any appropriate manipulation of the format of the messages. Likewise, HA servers 24 may manipulate messages from APS engine 22 that are to be communicated to external systems 40. To perform such manipulations, HA servers 24 may include a transform library 25 that is capable of re-packaging received messages into a format that can be interpreted by the component to which the message is to be communicated. Transform library 25 includes information regarding the appropriate format for external systems 40 and for APS engine 22, which may be updated if the type of external system 40 or APS engine 22 changes. Therefore, in one embodiment, APS engine 22 and external systems 40 do not need to be specially configured to send messages in a particular format.

HA systems 20 may also include a message mediator 26 that provides an interface between HA server 24 and an HA application program interface (API) 28. Message mediator 26 may receive queued messages from HA server 24, determine the types of operations requested in the messages, assign identifiers (IDs) to the requests, and queue the requests in an engine queue associated with APS engine 22. HA API 28 may than retrieve requests from the engine queue and translate the requests for processing by the associated APS engine 22. The functions and operation of HA server 24, message mediator 26, and HA API 28 are described in further detail below. However, it should be understood that although these exemplary components are described and illustrated, one or more other appropriate components may replace, combine, or cooperate with these components to enable communication of messages between external systems 40 and APS engines 22 of one or more HA systems 20.

HA systems 20 may include an administration tool 30 that is used by a system administrator to monitor the activity of HA system 20. Database system 60 may include a similar administration tool 66. Similarly, a Common Object Request Broker Architecture (CORBA) API 32a or other appropriate interface may be provided to allow an administrator to access data associated with APS engine 22a or any other appropriate component(s) of HA system 20. The various components of HA systems 20 and database system 60 may be implemented using any appropriate combination of hardware and/or software operating at one or more locations. Furthermore, the HA systems 20 in a particular HA system group may be located in one or more locations, and different HA system groups may be located in the same or different locations. In addition, external systems 40 may be located remotely from HA systems 20 and database system 60, as would typically be the case when message bus 50 incorporates the Internet.

As described above, certain embodiments of HA systems 20 include DF engines 22 that provide demand fulfillment services to external ordering systems 40. External ordering systems 40 may include any suitable systems that may be used to communicate requests to HA systems 20. For example, external ordering systems 40 may be used by customers to submit orders, inquiries, or other requests. A product orders is typically a request by a customer for a promise of product delivery, consistent with the order, from the entity associated with HA systems 20. Therefore, DF engines 22 are typically able to determine the supply of a product that is available to meet the customer order (the ATP supply) and to return a promise to the customer as to when its order can be fulfilled. DF engines 22 may also respond to customer inquiries, such as requests for quotations, regarding the availability of a certain product (for example, on a particular date, at a particular price, and at a particular location).

DF engines 22 may be used in conjunction with one or more demand planning engines. These demand planning engines may perform master demand planning that determines what demand has been committed to (what product orders have been promised) over a certain time horizon and determines the amount of supply that is ATP during this time horizon. Such a determination may be made using information about the manufacturing process used to create the product, the supply chain

supporting this manufacturing process, and/or any other appropriate information. As an example only, master demand planning may be performed at the end of each day to determine the amount of supply that is ATP for the next day (the ATP supply). However, the ATP supply may be determined at any other appropriate time intervals.

In certain embodiments, the ATP supply information is communicated to DF engine 22 in each HA system 20 in an HA system group before that system group goes on-line to become the operating HA system group. The ATP supply information may be stored by each HA system 20 in an associated memory or other appropriate data storage device. ATP supply information may also be stored in database 64 or any other appropriate remote location; however, each HA system 20 preferably uses locally stored, in-memory ATP supply information to improve the processing time associated with each transaction. HA systems 20 use this ATP supply information to process product orders, inquiries, and any other appropriate requests from external ordering systems 40.

As described above, in particular embodiments, primary HA system 20a is responsible for providing primary services to the external systems 40 and secondary HA systems 20b-20n are responsible for providing secondary services and fault protection. As an example only, the primary services provided by primary HA system 20a with an associated DF engine 22a may include processing customer orders for products from external ordering systems 40 and generating promises in response to these orders. In general, the primary services involve transactions that change the state of an HA system 20. For example, a promise generated by primary HA system 20a that allocates a certain amount of the ATP supply to a customer is a state-changing transaction since the ATP supply has changed and needs to be updated. Such state-changing transactions are significant, particularly in embodiments where such information is stored locally at each HA system 20, in that the ATP supply information (or other state information that is changed) needs to be updated at each HA system 20. As described below, primary HA system 20a replicates such state change information to secondary HA systems 20b-20n after completing a state-changing transaction. The performance of state-changing services may be limited to the primary HA system 20a to prevent multiple HA systems 20 from simultaneously changing the ATP supply and communicating such changes to one another.

Secondary services provided by secondary HA systems 20b-20n may include processing customer inquiries and generating responses to the inquiries (in addition to providing fault protection in the event primary HA system 20a fails or otherwise becomes unavailable). In general, the secondary services do not involve state changing transactions. For example, generating a response to an inquiry about the available supply does not require replication of data between HA systems 20 since there is no state change. Such inquiries may be automatically routed to secondary HA systems 20b-20n, leaving primary HA system 20a free to handle product orders and other state-changing requests. However, since secondary HA systems 20b-20n include the same or similar components as primary HA system 20a, one of the secondary HA systems 20b-20n may become the primary HA system 20a in the event the primary HA system 20a fails. In such a case, inquiry processing and other secondary services may be performed by the remaining secondary HA systems 20b-20n.

FIGURES 2A and 2B illustrates an exemplary method of processing orders using architecture 10. Though product orders are primarily described, the present invention contemplates orders for any suitable items. The method begins at step 100, where a product order is entered into an order management system 40 and communicated to an appropriate gateway 52. The order may be a request for a particular number of products on a particular date. At step 102, gateway 52 communicates the order as a request using message bus 50. As described above, primary HA system 20a may be used to process state-changing requests, such as product orders. Therefore, message manager 54 may direct state-changing requests to HA server 24a of primary HA system 20a. However, any HA system 20 may be used in other embodiments. HA server 24a receives the request at step 104 and preferably communicates an acknowledgement to gateway 52 using message bus 50.

HA server 24a uses transform library 25a at step 106 to parse the incoming order to examine the order for size and special characters, and communicates the order as a request to message mediator 26a at step 108 (the request may be "communicated" by placing the order in an internal request queue). HA server 24a then returns to receive other messages from gateways 52 at step 100, as indicated by arrow 110. At step 112, message mediator 26a retrieves the request from the request

queue (or otherwise receives the request) and identifies metadata in the request. The metadata includes parameterized operations that external ordering systems 40 can potentially invoke within a DF engine 22 using architecture 10. Therefore, the metadata includes information regarding the semantics that should be used to communicate data to and obtain data from an HA server 24a. In addition, the metadata may include parameters or tokens that inform HA server 22a to communicate with HA servers 24b-24n of secondary HA systems 20b-20n for replication of information and with HA server 62 of database system 60 for persistent storage of information.

At step 114, message mediator 26a attaches an internal request ID to the request and communicates the request to HA API 28a for initiation of order processing. This "communication" may be performed by placing the request in an internal DF engine queue. Message mediator 26a may also store the request ID in a second internal queue called a pending queue. HA API 28a receives the request (for example, HA API 28a may retrieve the request from the DF engine queue) and translates the request for processing by DF engine 22a at step 116.

DF engine 22a processes the request at step 118 by reviewing the ATP supply and generating a promise that includes a commit date, sourcing location(s), and/or other appropriate information. In connection with generating the promise, DF engine 22a may reserve ATP supply for the order and/or change internal allocations of ATP supply. HA API 28a communicates a response representing the promise to message mediator 26a at step 120. For each promise generated by DF engine 22a, HA API 28a may communicate a response including at least three components: (1) the promise to be communicated to the external system 40, (2) a replication message for secondary HA systems 20b-20n, and (3) a persistence message for database system 60. These components may be communicated together or separately to message mediator 26a (for example, each component may be communicated in a different packet). The replication message and the persistence message include information that allows secondary HA systems 20b-20n and database system 60, respectively, to determine what was promised by DF engine 22a (or otherwise identify a state change).

At step 122, message mediator 26a receives the response from HA API 28a, removes the request ID from the response (which was initially attached by message

mediator 26a), and attaches a transaction ID and time stamp to the response. The transaction ID and time stamp may be attached to each component of the response, if appropriate. Message mediator 26a may also perform a search on the pending queue, identify a matching request ID in the queue, and delete the request ID from the queue (to indicate that the request has been processed). Message mediator 26a communicates the response to HA server 24a at step 124.

At step 126, HA server uses transform library 25a to modify the format of the response as appropriate for the particular destination to which the response is to be communicated. For example, transform library 25a may modify the promise from DF engine 22 to a format that is appropriate for the destination external system 40. For instance, transform library 25a may change the character or line formatting, the type of units used to express certain numerical values (for example, currency type, product amounts, or measurement units), or any other appropriate information. As described above, use of transform library 25a may be advantageous since DF engine 22 will not have to be configured to produce a certain type of output for each different external system 40.

Transform libraries 25 may also be used to modify the format of a request from an external system 40 as appropriate for different versions of an APS engine 22. For example, an old version and a new version of an APS engine 22 (in the same or different HA systems 20) may be used to process the same product orders until an administrator is satisfied that the new version is operating properly (the results from the two versions may be compared to ensure proper operation of the new version). In such a situation, a copy of the product order may communicated to each version of the APS engine 22 in a format used by the old version. The transform library 25 associated with the new version may be used to modify incoming requests that are formatted for the old version to account for a different type of formatting required by the new version. Once the old version is replaced by the new version, external systems 40 may be instructed to change the formatting of their requests to comply with the new version or transform libraries 25 may continue to be used to modify the format of incoming requests.

Particular embodiments may also incorporate an engine version number in each incoming request. In such embodiments, an external system 40 (or different

external systems 40) may send multiple requests that are each formatted for a different engine version and may specify the version number in the requests. The messages may be directed by message manager 54 or any other appropriate component to the appropriate HA system 20 based on the engine version used by that HA system 20. This use of version numbers may be useful when testing and implementing a new engine version. For example, while the new version is being tested, an external system 40 may be directed to communicate a particular request in a first format appropriate for the new version and in a second format appropriate for the old version. The version number included each the requests may then be used to appropriately direct the requests. Once the new engine version is tested and implemented, the external system 40 may be instructed to only send requests in the first format.

Returning to the exemplary method of FIGURES 2A and 2B, at steps 128a-128c (which may be performed substantially simultaneously or in any appropriate order), HA server 24a communicates the promise to the appropriate gateway 52, communicates the replication message to secondary HA systems 20b-20n, and communicates the persistence message to database system 60 using message bus 50. Gateway 52 communicates the promise to the appropriate external ordering system 40 at step 130a. At step 130b, HA systems 20b-20n process the replication message such that the ATP supply information or other appropriate state information associated with DF engines 22b-22n is changed to duplicate the state of DF engine 20a after processing the order. HA API 28 of each HA system 20b-20n may initiate the appropriate action by DF engines 22b-22n to replicate the state of DF engine 22a. For example, DF engine 22a may modify the ATP supply that may be promised to external systems 40 to reflect a promise of ATP supply generated by DF engine 22a. Modification of ATP supply information stored at DF engines 22b-22n may be in response to information in the replication message and not due to a true re-processing by DF engines 20b-20n of the product order that was originally communicated to primary HA system 20a. This may be referred to as "replaying" the product order. Therefore, the processing power of secondary HA systems 20b-20n is not unnecessarily used. At step 130c, HA server 62 receives the persistence message (which may include information about the product order, the replication performed,

and the promise) and communicates the information to database 64 for storage. The exemplary method may be repeated for each product order or other state-changing request generated by an external ordering system 40.

While an exemplary method is illustrated and described, architecture 10 contemplates using any suitable techniques and components for communicating product orders to DF engine 22, processing these orders, communicating responses to external ordering systems 40, and replicating appropriate information. Moreover, certain steps in this method may take place substantially simultaneously and/or in different orders than as described. Architecture 10 also contemplates using other appropriate methods with additional steps, fewer steps, or different steps.

As described above, architecture 10 provides fault protection mechanisms that allow a secondary HA system 20b-20n to replace primary HA system 20a in the event of primary HA system 20a fails or otherwise becomes unavailable. In such an event, the failing primary HA system 20a communicates an alert message to one of the secondary HA engines 20b-20n (HA system 20b in this example). In addition to or instead of such a notification, message manager 54 or any other appropriate component may poll HA systems 20 to determine when an HA system 20 has failed. Message manager 54 may then notify secondary HA system 20b of the failure of primary HA system 20a. Alternatively, HA systems 20 may poll one another. Upon receipt of a failure notification regarding primary HA system 20a or otherwise determining that primary HA system 20a has failed, secondary HA system 20b registers itself with message manager 54 as the new primary HA system 20a causing message manager 54 to direct subsequent state-changing requests to the new primary HA system 20a. Since the previous primary HA system 20a replicated state changing information to the new primary HA system 20a when it was secondary HA system 20b, there is typically no loss of pending requests or inconsistency in state information upon the changeover.

FIGURE 3 illustrates an exemplary method of processing product inquiries using architecture 10. The method begins at step 150, where a product inquiry is entered into an order management system 40 and communicated to an appropriate gateway 52. For example, the inquiry may include a query as to whether a particular amount of a product is available for delivery on a particular date. At step 152,

gateway 52 communicates the inquiry as a message using message bus 50. As described above, secondary HA systems 20b-20n are preferably used to process non-state-changing requests, such as product inquiries. Therefore, message manager 54 may direct non-state-changing requests to HA server 24 of any secondary HA system 20b-20n. This frees up primary HA system 20a in embodiments in which primary HA system 20a processes all state-changing requests. However, in other embodiments, any suitable HA system 20 may be used.

Communicating inquiries and other non-state-changing requests to secondary HA systems 20b-20n instead of primary HA system 20a serves to balance the load between the primary HA system 20a and the secondary HA systems 20b-20n. However, at least in one embodiment, primary HA system 20a may also process non-state-changing requests if appropriate in view of the number of state-changing requests being processed at that time. In addition, message manager 54 may perform load balancing between each of the secondary HA systems 20b-20n by directing each inquiry (or other non-state-changing request) to a particular secondary HA system 20b-20n based on the number of inquiries that are in the request queue of that and possibly each secondary HA system 20b-20n (or based on any other indication of the processing load of that secondary HA system 20b-20n, viewed in isolation or relative to other secondary HA systems 20b-20n). Message manager 54 may communicate with an HA server 24 to determine the number of inquires in its request queue, may communicate with a daemon or other message manager associated with each secondary HA system 20b-20n, or may use any other appropriate method to determine the current load on one or more secondary HA systems 20b-20n. Message manager 54 may then determine at step 152 which secondary HA system 20b-20n has the least amount of load and direct the inquiry to the HA server 24 of that secondary HA system 20b-20n.

At step 154, an appropriate HA server 24 (for example, HA server 24b) receives the request and preferably communicates an acknowledgement to gateway 52 using message bus 50. HA server 24b uses transform library 25b at step 156 to parse the incoming inquiry to examine the inquiry for size and special characters, and communicates the inquiry as a request to message mediator 26b at step 158 (again, the request may be "communicated" by placing the inquiry in an internal request queue).

HA server 24b then returns to receive other messages from gateways 52 at step 150, as indicated by arrow 160. At step 162, message mediator 26b retrieves the request from the request queue (or otherwise receives the request) and identifies metadata in the request as described above with reference to FIGURES 2A and 2B. At step 164, message mediator 26b attaches an internal request ID to the request and communicates the request to HA API 28b for initiation of inquiry processing. This "communication" may be performed by placing the request in an internal DF engine queue. Message mediator 26b may also store the request ID in a pending queue. HA API 28b receives the request (for example, HA API 28b may retrieve the request from the DF engine queue) and translates the request for processing by DF engine 22b at step 166.

DF engine 22b processes the request at step 168 by reviewing the ATP supply and generating a response that may include possible commit dates, possible sourcing location(s), and/or other appropriate information. Since the response to the inquiry is not a promise of ATP supply, DF engine 22b does not need to change the ATP supply information. HA API 28b communicates the response to message mediator 26a at step 170. At step 172, message mediator 26b receives the response from HA API 28a, removes the request ID from the response (which was initially attached by message mediator 26a), and attaches a transaction ID and time stamp to the response. Message mediator 26b may also perform a search on the pending queue, identify a matching request ID in the queue, and delete the request ID from the pending queue (to indicate that the request has been processed). Message mediator 26b communicates the response to HA server 24b at step 174. At step 176, HA server 24b uses transform library 25b to modify the format of the response as appropriate for the particular external ordering system 40 to which the response is to be communicated, as described above. At step 178, HA server 24b communicates the response to the appropriate gateway 52 and gateway 52 communicates the response to the appropriate external ordering system 40 at step 180.

While an exemplary method is illustrated and described, architecture 10 contemplates using any suitable techniques and components for communicating product inquiries to DF engine 22, processing these inquiries, and communicating responses to external ordering systems 40. Moreover, certain steps in this method

may take place substantially simultaneously and/or in different orders than as described. Architecture 10 also contemplates using other appropriate methods with additional steps, fewer steps, or different steps.

FIGURE 4 illustrates an operating HA system group 70a and a replacement HA system group 70b included in architecture 10. Although multiple HA systems 20 and 20' are included in each system group 70, in particular embodiments each system group 70 may only include one HA system 20 or 20'. As described above, ATP supply information and/or other appropriate planning information generated by one or more planning engines 80 may be communicated to the DF engine 22 in each HA system 20 in replacement HA system group 70b before that system group goes on-line to become the operating HA system group 70a. Planning engine 80 may generate such planning information on a periodic or other suitable basis. As an example only, planning engine 80 may perform master demand planning at the end of a day to determine the amount of ATP supply that may be promised by HA systems 20 during the next day (based in part on the amount promised by HA systems 20 during the previous day). When this ATP supply information or other planning information is generated, the information currently stored in memory by HA systems 20 in operating HA system group 70a needs to be updated. Since it is preferable to perform this updating while HA systems 20 in HA system group 70a are off-line, architecture 10 provides a mechanism to allow operating HA system group 70a to be seamlessly replaced by replacement HA system group 70b, which has already received and stored the updated planning information from planning engine 80.

FIGURE 5 illustrates an exemplary method of updating planning information and synchronizing that planning information between an operating HA system 20a and a replacement HA system 20a'. As stated above, although the exemplary method includes the use of planning engine 80 to perform demand planning and to generate ATP supply information for use by DF engines 22, the present invention encompasses the use of any appropriate APS engines in conjunction with any appropriate type of planning activity. The exemplary method begins at step 200, where planning engine 80 extracts change information from database system 60 and/or operating HA system group 70a. The change information may include actual planning information, such as ATP supply information, or information reflecting changes made to the planning

information by HA systems 20 over a certain time period. The time at which the extraction takes place is stored in a synchronization table in database 64 or in any other suitable format and/or location. In certain embodiments, planning engine 80 may extract change information associated with subsets of the planning information. For example, planning engine 80 may extract planning information associated with requests for products available from a particular supplier or planning information associated with a particular type of request. In such cases, an extraction time is stored for each subset that is extracted.

At step 202, planning engine 80 determines the difference between the amount of demand forecasted by planning engine 80 and the actual demand for a time period (for example, based on the amount of the ATP supply that was promised during the previous period by HA systems 20 as determined from the change information). Based on this demand information, planning engine 80 updates the amount of ATP supply for the next period at step 204 and communicates the updated ATP supply information to HA systems 20' of replacement HA system group 70b at step 206. Primary HA system 20a of operating HA system group 70a continues to promise ATP supply in response to product orders while the ATP supply information is being updated. Therefore, once replacement HA system group 70b is ready to come on-line, the ATP supply information stored in replacement HA systems 20' should be updated to account for promises made by operating primary HA system 20a after planning engine 80 extracted the ATP supply information.

When replacement HA system group 70b is ready to come on-line, DF engine 22a' of replacement primary HA system 20a' requests a synchronization service from HA server 62 of database system 60 at step 208. HA server 62 determines what product orders were processed by primary HA system 20a after the extraction time ("post-extraction orders") at step 210. HA server 62 may identify the post-extraction orders by determining the extraction time and searching for processed product orders in database 64 (communicated from operating primary HA system 20a) having time stamps indicating that the orders were processed after the extraction. At step 212, HA server 62 communicates the post-extraction product orders (or other suitable information representing the content of the product orders) to replacement primary HA system 20a'.

At step 214, primary HA system 20a' processes or replays the product orders or other appropriate information, as described above in relation to FIGURES 2A and 2B, and updates the ATP supply information stored at primary HA system 20a'. The products orders or other appropriate information may be processed in any suitable manner such that the ATP supply information at primary HA system 20a' is properly updated. Any promises that may be generated by this processing (as described in relation to FIGURES 2A and 2B) are not communicated to external ordering systems 40 since such promises would already have been communicated by operating primary HA system 20a. However, primary HA system 20a' may communicate replication messages to secondary HA systems 20b'-20n' so that these systems may also update their ATP supply information.

Replacement primary HA system 20a' instructs operating HA systems 20 (either individually or through primary HA system 20a) to terminate operation at step 216. At step 218, HA systems 20 terminate operation and generate a stop record indicating the last order that was processed and/or the time of the termination. At substantially the same time as the post-extraction orders are processed by replacement primary HA system 20a' and the termination order is sent, HA systems 20' become operational at step 220 and take over for HA systems 20 (for example, become ready to process product orders, inquiries, and other appropriate requests). It is possible that primary HA system 20a may process a small number of product orders during or after the time that the stop record is published (for example, product orders that were already being processed when primary HA system 20a was instructed to terminate operation). Therefore, at step 222 the new operating primary HA system 20a' may process or replay (as in step 214) any remaining pre-termination orders that were already processed by primary HA system 20a and update the HA supply information in ATP systems 20' accordingly.

While an exemplary method is illustrated and described, architecture 10 contemplates using any suitable techniques and components for seamlessly replacing an operating HA system group with a replacement HA system group. Moreover, certain steps in this method may take place substantially simultaneously and/or in different orders than as described. Architecture 10 also contemplates using other appropriate methods with additional steps, fewer steps, or different steps.

In summary, the present invention provides an architecture 10 that provides fault tolerant, real-time communication between multiple APS engines 22 and one or more external systems 40. The multiple APS engines 22 interact such that if one APS engine 22 fails or otherwise become unavailable, another APS engine 22 can seamlessly take over for that APS engine 22. Furthermore, the multiple APS engines 22 allow for load balancing through assignment of different types of requests to different APS engines 22 and/or allocation of multiple requests of a particular type between multiple APS engines 22. In addition, requests from external systems 40 may be transformed as appropriate for a particular APS engine 22 and/or a response from an APS engine 22 may be transformed as appropriate for a particular external system 40. Therefore, APS engines and external systems may not need to be specially configured before being included in architecture 10. Moreover, a mechanism may be provided by which the planning information stored in APS engines 22 may be updated while still providing uninterrupted service to external systems 40.

Although the present invention has been described with several embodiments, numerous changes, substitutions, variations, alterations, and modifications may be suggested to one skilled in the art, and it is intended that the invention encompass all such changes, substitutions, variations, alterations, and modifications as fall within the spirit and scope of the appended claims.